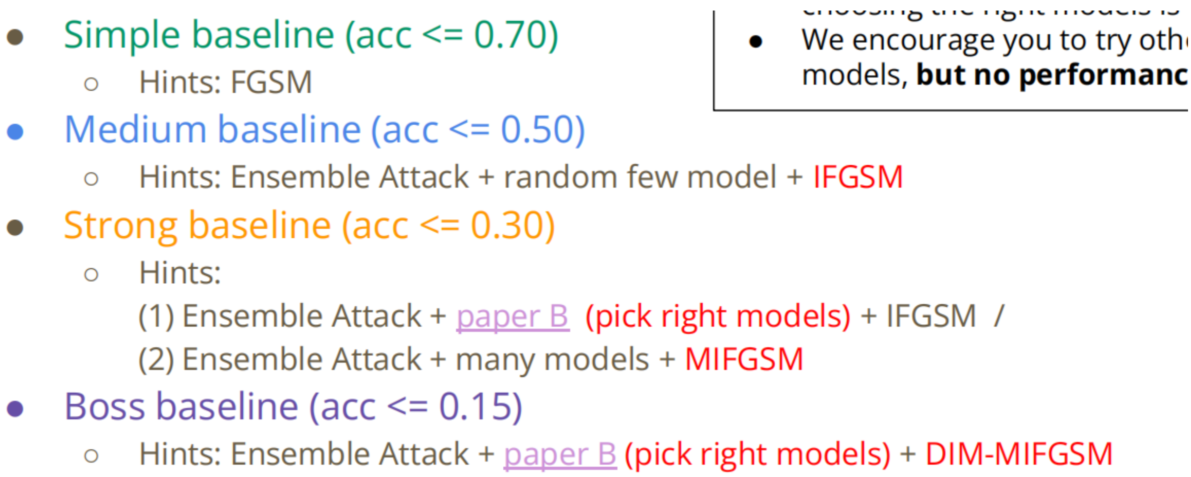
HW10

1. 作业要求

HW10 是做对抗性攻击的，首先使用 pytorchcv 来获得 CIFAR-10 预训练模型，下载想要攻击的数据(200 张图片)。然后使用一些攻击算法在代理网络上进行攻击，使用的攻击模式是黑盒攻击（对代理网络执行攻击），具体的作业要求如下图所示。



- Simple baseline (acc <= 0.70)
  - Hints: FGSM
- Medium baseline (acc <= 0.50)
  - Hints: Ensemble Attack + random few model + IFGSM
- Strong baseline (acc <= 0.30)
  - Hints:
    (1) Ensemble Attack + paper B (pick right models) + IFGSM /
    (2) Ensemble Attack + many models + MIFGSM
- Boss baseline (acc <= 0.15)
  - Hints: Ensemble Attack + paper B (pick right models) + DIM-MIFGSM

choosing the right models is

- We encourage you to try oth
models, **but no performanc**

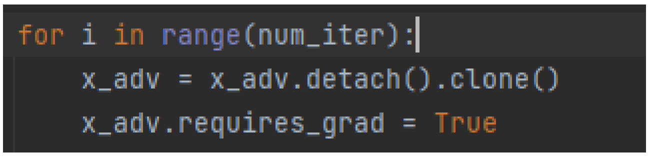
2. 实验相关

实验相关的主要是几个攻击算法，分别是 FGSM,IFGSM,MIFGSM,DIM-MIFGSM.

最简单的是 FGSM（Fast Gradient Sign Method）,是一种基于梯度生成对抗样本的算法，属于对抗攻击中的无目标攻击（即不要求对抗样本经过 model 预测指定的类别，只要与原样本预测的不一样即可）。我们在理解简单的 dp 网络结构的时候，在求损失函数最小值，我们会沿着梯度的反方向移动，使用减号，也就是所谓的梯度下降算法；而 FGSM 可以理解为梯度上升算法，也就是使用加号，使得损失函数最大化。SIGN 函数用于返回数字的符号。当数字大于 0 时返回 1，等于 0 时返回 0，小于 0 时返回 -1。 x 是原始样本，θ 是模型的权重参数（即 w），y 是 x 的真实类别。输入原始样本，权重参数以及真实类别，通过 J 损失函数求得神经网络的损失值，∇ x 表示对 x 求偏导，即损失函数 J 对 x 样本求偏导。ϵ （epsilon）的值通常是人为设定 ，可以理解为学习率，一旦扰动值超出阈值，该对抗样本会被人眼识别。

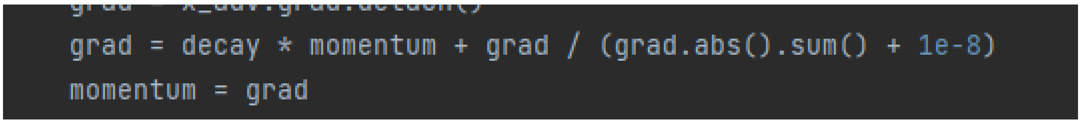
```
def fgsm(model, x, y, loss_fn, epsilon=epsilon1):
    x_adv = x.detach().clone()
    x_adv.requires_grad = True
    loss = loss_fn(model(x_adv), y)
    loss.backward()
    grad = x_adv.grad.detach()
    x_adv = x_adv + epsilon * grad.sign()
    return x_adv
```

接下来是 IFGSM，在 FGSM 的基础上增加迭代次数，虽然计算成本会增加，但是最坏的结果也是和原始的 FGSM 一样。



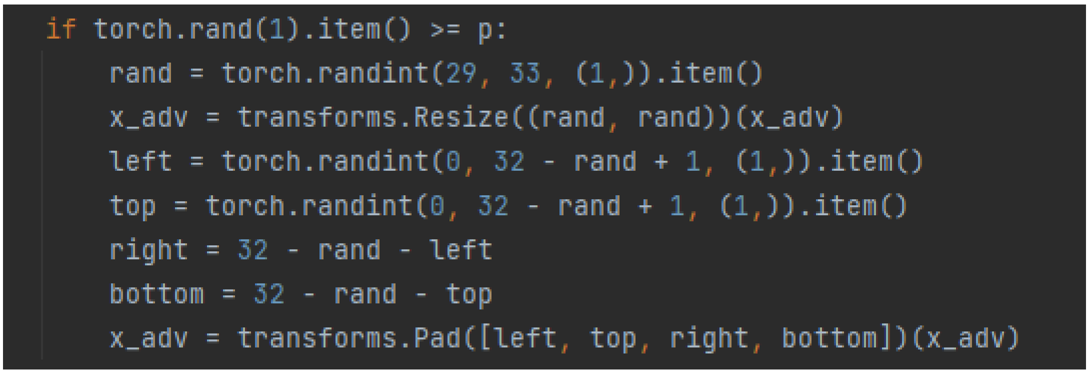
```
for i in range(num_iter):
    x_adv = x_adv.detach().clone()
    x_adv.requires_grad = True
```

MIFGSM 是增加动量，类似于学习率里面的动量，用动量增强对抗性攻击，使用动量来稳定更新方向和逃避糟糕的局部最大值。



```
grad = x_adv.grad.detach()
grad = decay * momentum + grad / (grad.abs().sum() + 1e-8)
momentum = grad
```

对于 DIM-MIFGSM 是在 MIFGSM 的基础上增加多种输入，也就是 DIM，随机调整大小，随机填充（以随机的方式在输入图像周围填充零）



```
if torch.rand(1).item() >= p:
    rand = torch.randint(29, 33, (1,)).item()
    x_adv = transforms.Resize((rand, rand))(x_adv)
    left = torch.randint(0, 32 - rand + 1, (1,)).item()
    top = torch.randint(0, 32 - rand + 1, (1,)).item()
    right = 32 - rand - left
    bottom = 32 - rand - top
    x_adv = transforms.Pad([left, top, right, bottom])(x_adv)
```

3. 作业结果

作业里面按照 boss 的要求，使用 DIM-MIFGSM，使用 https://github.com/kuangliu/pytorch-cifar 中的 resnet18, 这个训练两百个 epoch 可以达到较好的结果，所以在训练时选择 50 次，训练的不那么好，用做 ensembleNet 里面，来进行攻击，最终各个算法的结果如下所示，依次是 IFGSM,MIFGSM,DMI-MIFGSM。

benign: airplane1.png
airplane: 27.85%

adversarial: airplane1.png
ship: 20.59%

benign: automobile1.png
automobile: 36.25%

adversarial: automobile1.png
automobile: 16.47%

benign: bird1.png
bird: 25.12%

adversarial: bird1.png
dog: 17.08%

benign: cat1.png
cat: 49.15%

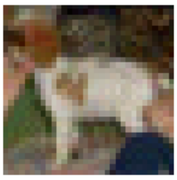
adversarial: cat1.png
cat: 27.21%

benign: deer1.png
deer: 30.23%

adversarial: deer1.png
frog: 14.52%

benign: dog1.png
dog: 18.73%

adversarial: dog1.png
cat: 36.20%



benign: frog1.png
frog: 23.79%

adversarial: frog1.png
deer: 22.59%

benign: horse1.png
horse: 35.87%

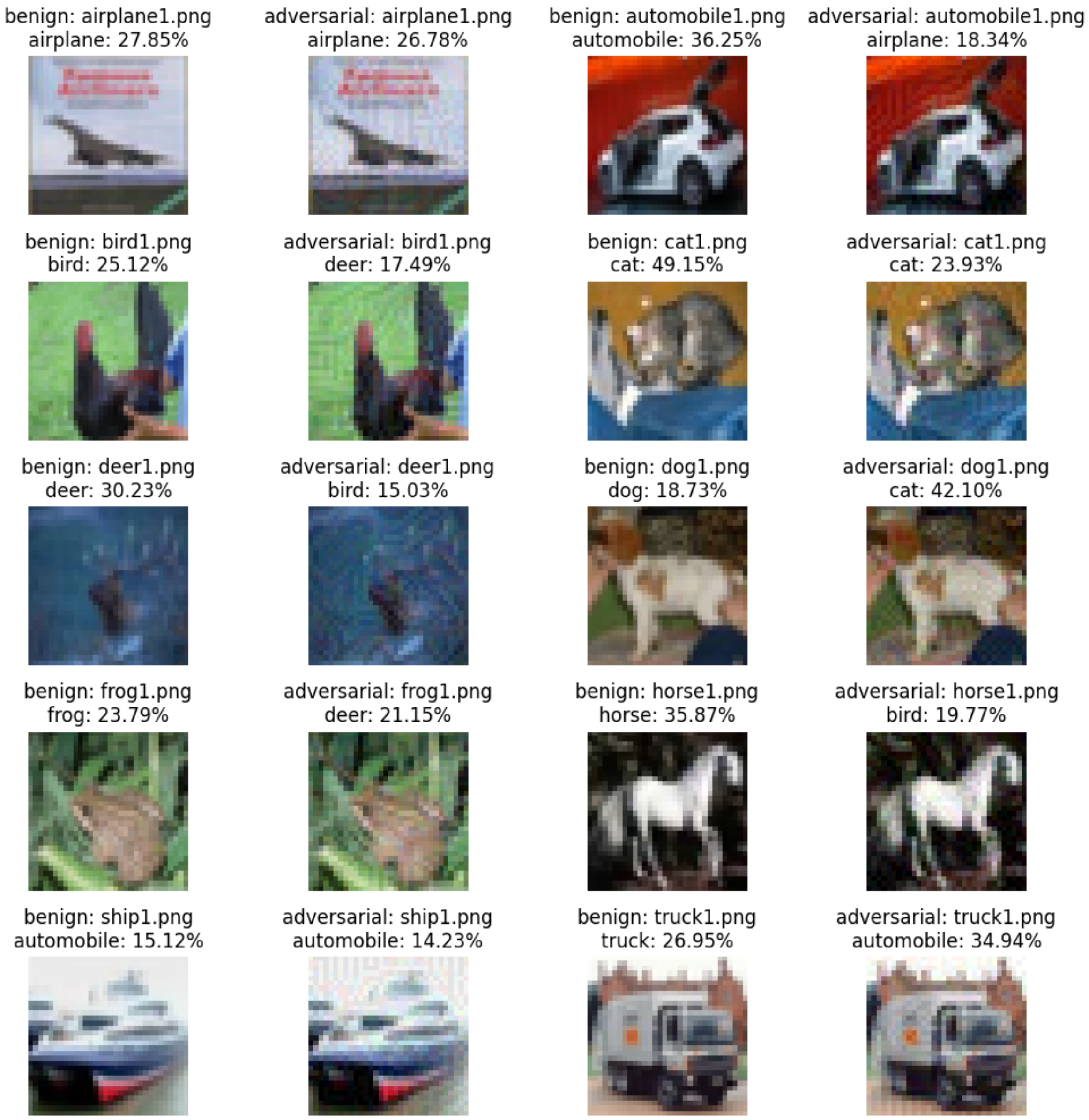
adversarial: horse1.png
bird: 16.74%

benign: ship1.png
automobile: 15.12%

adversarial: ship1.png
airplane: 13.91%

benign: truck1.png
truck: 26.95%

adversarial: truck1.png
automobile: 34.14%



benign: airplane1.png
airplane: 27.85%

adversarial: airplane1.png
airplane: 26.78%

benign: automobile1.png
automobile: 36.25%

adversarial: automobile1.png
airplane: 18.34%

benign: bird1.png
bird: 25.12%

adversarial: bird1.png
deer: 17.49%

benign: cat1.png
cat: 49.15%

adversarial: cat1.png
cat: 23.93%

benign: deer1.png
deer: 30.23%

adversarial: deer1.png
bird: 15.03%

benign: dog1.png
dog: 18.73%

adversarial: dog1.png
cat: 42.10%

benign: frog1.png
frog: 23.79%

adversarial: frog1.png
deer: 21.15%

benign: horse1.png
horse: 35.87%

adversarial: horse1.png
bird: 19.77%

benign: ship1.png
automobile: 15.12%

adversarial: ship1.png
automobile: 14.23%

benign: truck1.png
truck: 26.95%

adversarial: truck1.png
automobile: 34.94%

benign: airplane1.png
airplane: 27.85%

adversarial: airplane1.png
airplane: 24.14%

benign: automobile1.png
automobile: 36.25%

adversarial: automobile1.png
cat: 30.63%

benign: bird1.png
bird: 25.12%

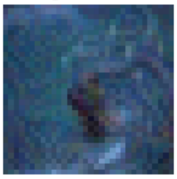
adversarial: bird1.png
cat: 21.26%

benign: cat1.png
cat: 49.15%

adversarial: cat1.png
cat: 26.07%

benign: deer1.png
deer: 30.23%

adversarial: deer1.png
frog: 16.28%



benign: dog1.png
dog: 18.73%

adversarial: dog1.png
cat: 34.25%

benign: frog1.png
frog: 23.79%

adversarial: frog1.png
deer: 24.00%

benign: horse1.png
horse: 35.87%

adversarial: horse1.png
bird: 15.64%

benign: ship1.png
automobile: 15.12%

adversarial: ship1.png
automobile: 17.97%

benign: truck1.png
truck: 26.95%

adversarial: truck1.png
automobile: 26.96%